محمد امامی   ۸۱۰۱۰۰۰۸۴   تکلیف اول کامپایلر   استاد توسلی

۱) الف) ۱- در زبان طبیعی ما جملات imperative را جملات دستوری که فاعل آن محذوف است و جملات declarative را جملات بیانی که حقیقتی را با بیان فاعل بیان می‌کنند می‌نامیم
مانند در programming languages به اشکال یک دستور declarative باعث تغییر value یک variable نمی‌شود اما یک expression که imperative است یک variable را با مقدار بیشتر تغییر می‌دهند (تغییر مقدار متغیر)

۲- در واقع اگر به جای یک متغیر مثلاً x متغیری دیگر را در آن scope قرار دهیم نباید تغییری در مقادیر ما ایجاد شود یا به عبارت دیگر در یک scope اگر ما تعداد معینی متغیر تعریف شده مثل x_1 تا x_n داشته باشیم و یک expression دلخواه مثل e را در نظر بگیریم که شامل این متغیرها (x_1 تا x_n) باشد، مقدار این expression در همه جای scope مقدار ثابتی دارد.

۳- به دلیل آنکه در زبان‌های pure-functional، expression ها همگی مستقل هستند و این زبان‌ها زبان‌های Declarative هستند، پس به دلیل عدم مزاحمت و ارتباط بین expression ها می‌توان اجرای همه expression ها را به صورت parallel انجام داد. در واقع وقتی تابعی مثل f با ورودی‌های e_1 تا e_n را call می‌کنیم، چون e_1 تا e_n مستقل هستند و خروجی یکی به دیگری وابسته نیست پس می‌توانیم این پارامترها را به صورت موازی محاسبه کنیم.

۴- همچنین این امر موجب بهینه‌سازی [redacted] می‌شود علی‌الخصوص بهینه‌سازی در Resource ها. چرا که در صورتی که اجرای یک expression به تغییری دیگری وابسته باشد یعنی if e_1 then e_2 else e_3 رخ دهد. حال محاسبه e_1، e_2، e_3 به صورت همزمان یعنی محاسبه اضافی دست کم یکی از این expression ها و سپس نیازی نیست که هر دوی e_2، e_3 را انجام دهیم که باعث هدر رفت منابع و محاسبات اضافی می‌شود.

ب) Fibonacci.

```
fun Declarative-fib(n : int)
        if n >= 2
            return (Declarative-fib(n-1) + Declarative-fib(n-2))
    else
        return 1;
```

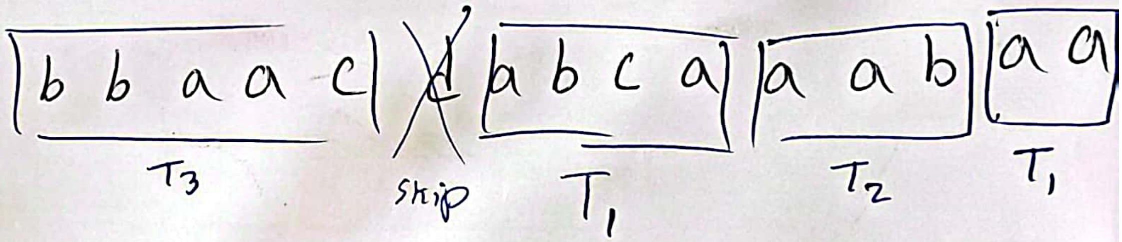
```
fun Imperative_fib(n, int)
    int num1 = 1;
    int num2 = 1;
    for( int i=2; i<n; i++){
        if (min(num1,num2) == num1)
            num1 = num1 + num2
        else if (min(num1,num2) == num2)
            num2 = num1 + num2
    }
    return max(num1,num2)
```

---

(2)

$T = T_1 + T_2 + T_3$

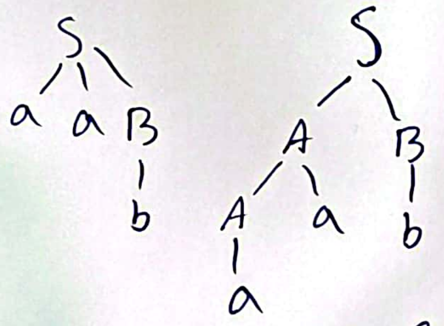
در این روش (panic mode) باید آنقدر skip کنیم تا با یک prefix match شود. (d را skip می‌کنیم)



---

(3)

حال ۲ اشتقاق داریم پس مبهم است.



حال باید رفع ابهام کنیم.

برای رفع ابهام یک عبارت بی‌تاثیری مثل $S \to aaB$ را حذف می‌کنیم بدون اینکه زبان عوض شود. چرا که تمامی رشته‌های شامل aaB با استفاده از ۲ بار A و یکبار B قابل ایجاد هستند. (درخت بالا)

پس

$S \to AB$

$A \to a \mid Aa$

$B \to b$

4 (الف)

$S \rightarrow asasb \mid asbsa \mid bsasa \mid SS \mid \varepsilon$

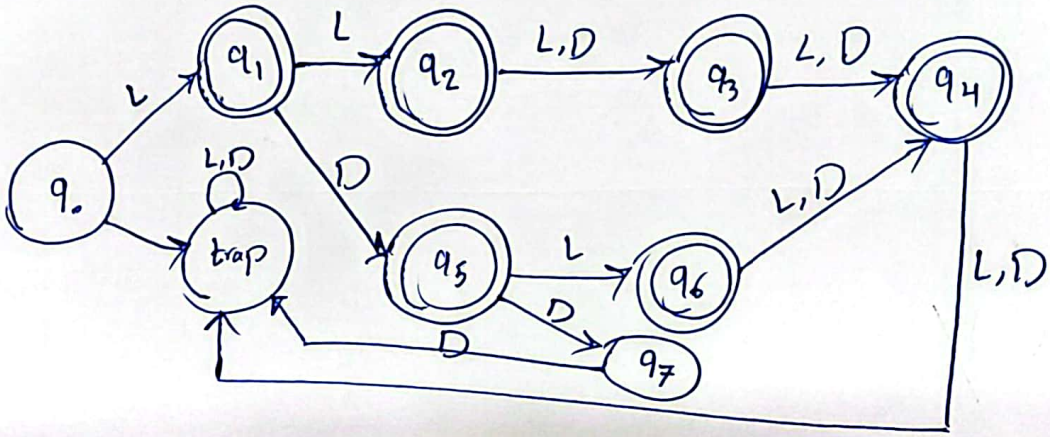
(ب)

$S \rightarrow [\,] \mid [\,E\,]$

$E \rightarrow S \mid 0 \mid 1 \mid E,E$

---

5 (الف)



(ب)

Letters = [a-z] (حروف کوچک انگلیسی)

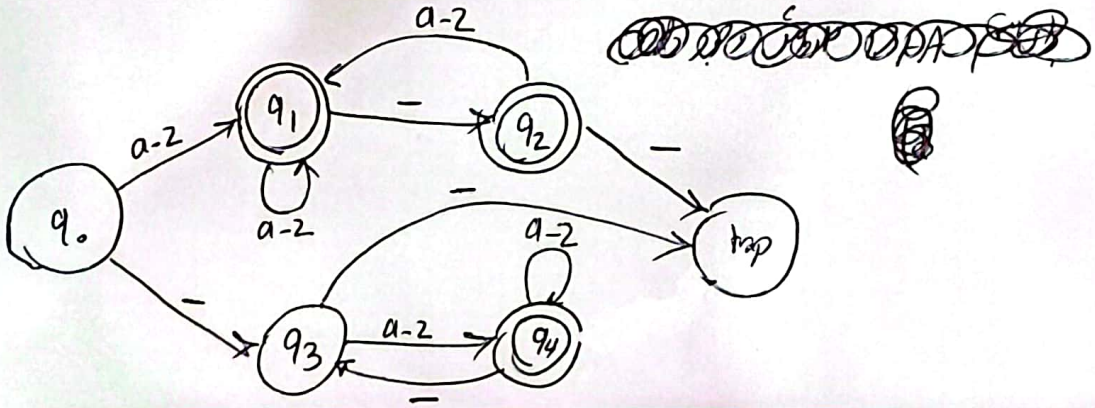
Parser = —

$\Rightarrow$ Private = $(USS^*)(USS^*)^* \longrightarrow (US^+)(US^+)^* =$

$(US^+)^+ \rightarrow$ Private

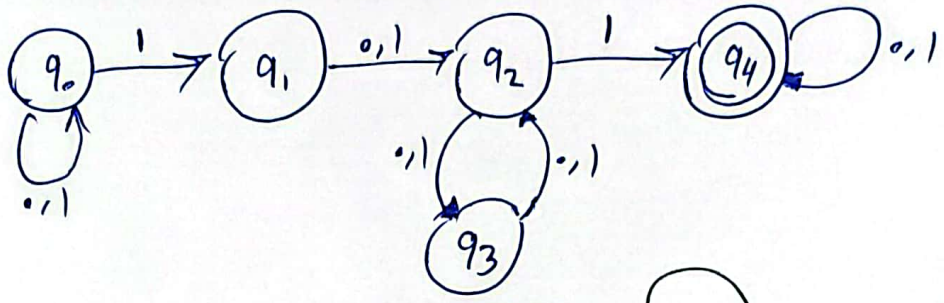
Public = $S^+(US^+)^*(U+\varepsilon)$

L = Private + Public

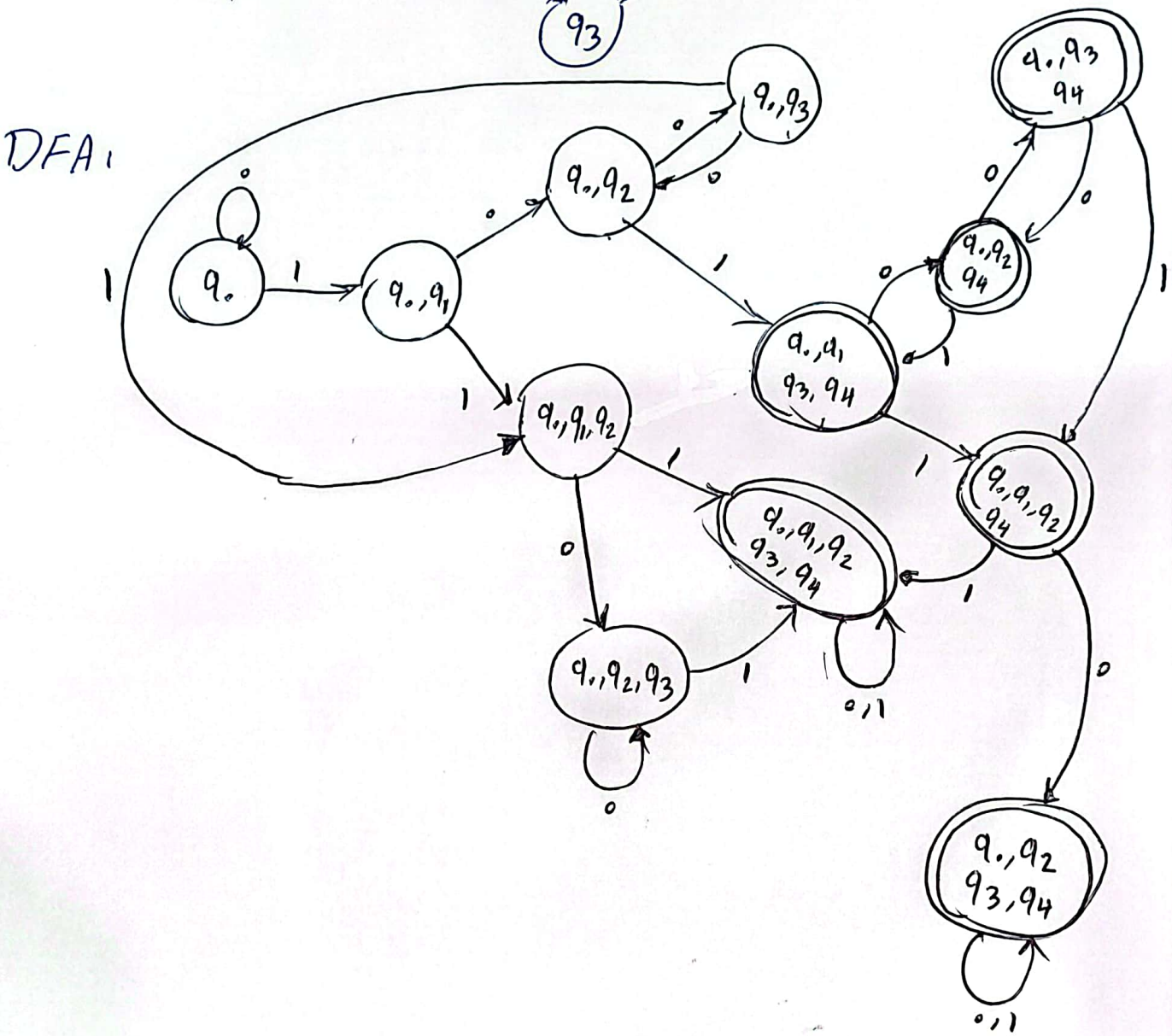


$(0+1)^*1(0+1)((0+1)(0+1))^*1(0+1)^*$ 12.

NFA



DFA:



Digits = [0-9]

Letters = [0-9, A-F]

$$R_{Dec} = b-(0+1)^* + h-(Letters)^* + (d-+\varepsilon)(Digits)^*$$

$$R_{Octal} = b-(0+1)^*.(0+1)^* + h-(Letters)^*.(Letters)^* + (d-+\varepsilon)(Digits)^*.(Digits)^*$$

$$R_{Binary} = R_{Dec}f$$

$$R = R_{Dec}f + R_{Dec} + R_{Octal}$$